InkBridge Networks

TEAPv2

EMU - IETF 124

WHAT'S WRONG WITH TEAPv1?

- Key derivation is complex. Start with

  $key_0 = TLS\text{-}Exporter(\ldots)$

- Then, for each round:

  $MSK\text{-}key_n = mash(MSK_n, key_{n-1})$ $EMSK\text{-}key_n = mash(EMSK_n, key_{n-1})$

  $key_n = choose???(MSK\text{-}key_n \text{ or } EMSK\text{-}key_n)$

- MSK and/or EMSK might not exist, and MSK is sometimes set to zero.
- Everyone did something different for the key derivation.

EVEN MORE “WHAT’S WRONG WITH TEAPv1”

- Crypto-Binding is tied to authentication methods
  - Except for Basic-Password
  - and not for other inner exchanges (CSR, etc.)
- There is a lot of uncertainty around portions of the protocol which have not been used
  - PKCS#7 and PKCS#10 have no implementations or use cases

WHAT WE WOULD LIKE TO DO

- It would be good to have interoperable implementations
  - before any new specification is published.
  - That everyone can test against
- Implement more features
  - so we know that they work

HOW TO FIX IT

- We really only need one initial crypto-binding exchange
  - If that works, we know that the tunnel is secure
- Do we really need to tie all inner methods together via Crypto-Binding TLV?
  - Because we don’t actually do that for TEAPv1
  - So we can likely just drop it for TEAPv2, as it adds significant complexity

POSSIBILITIES

- Note unimplemented features from TEAPv2
  - e.g. no PKCS#7 or PKCS#10
  - Do not deprecate them, but mark them as not required
- EAP-TLS shows that empty provisions are nearly meaningless
- We leave untested features to future RFCs

QUESTIONS

- Do we want to do this?
- What time frame?